COL 774: Machine Learning

Assignment 1

**BRAJ RAJ NAGAR**

**2022AIB2682**

[**aib222682@scai.iitd.ac.in**](mailto:aib222682@scai.iitd.ac.in)

1. **Linear Regression**

One of the most often used machine learning algorithms for learning linear models from data is linear regression. In order to predict the density of wine based on its acidity, In this problem we use a type of linear regression model known as least squares linear regression. The feature x ∈ IR is ”Acidity of the wine” and the output y ∈ IR is ”Wine Density”.

1. The least squared error function, often known as the cost function J(θ), in this problem was optimized using the batch gradient descent method as follows:

J(θ) =

Initialize the parameters as θ = (the vector of all zeros)

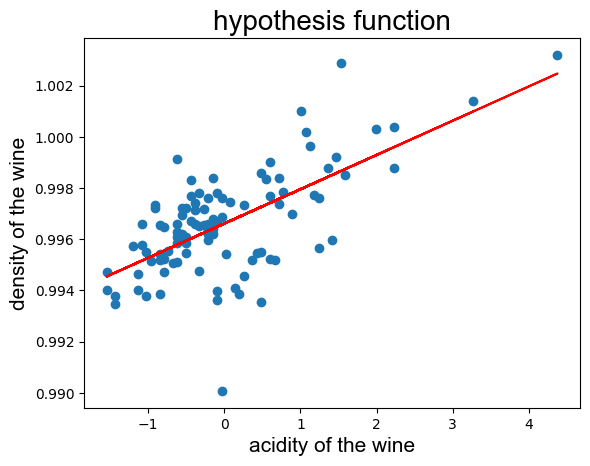
**Learning Rate: 0.1**

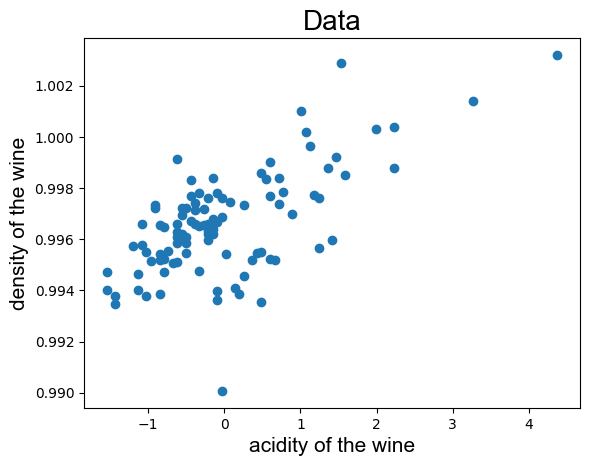
**Stopping Criteria is: current cost - previous cost < 1e-11**

**Final set of parameters obtained by algorithm:**

**= 0.9966108699154312 =0.0013401836064368222**

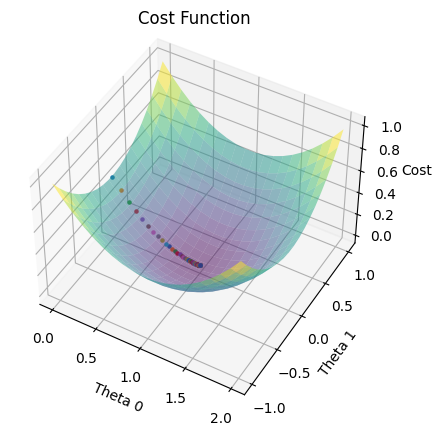
**b)**





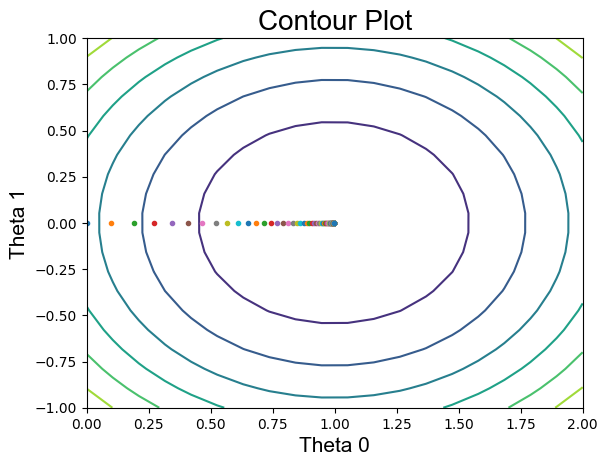
Data and for Linear Regression

**c)**

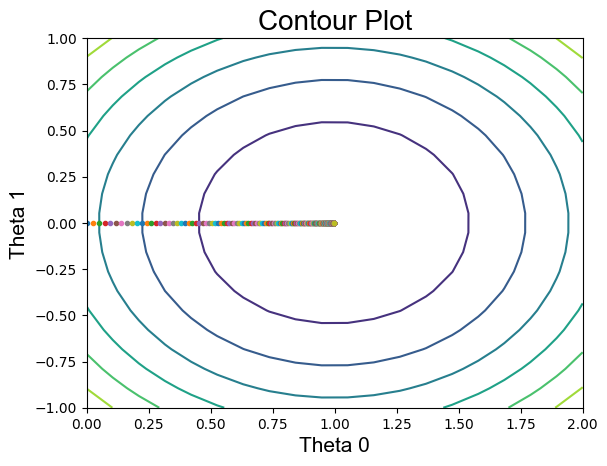


Cost Function plotted w.r.t to and

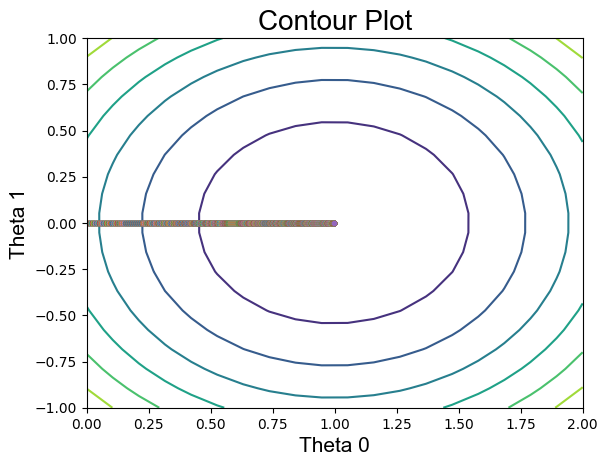
**d) and e)**



Contour at Learning Rate 0.1



Contour at Learning Rate 0.025



Contour at Learning Rate 0.001

If we examine the contour plots, we can see that the bigger the learning rate, the larger the steps that Gradient Descent takes to converge to optimum values of θ.

The Gradient Descent Algorithm will diverge after a certain value of learning rate, if we increase learning rate further.

1. **Sampling and Stochastic Gradient Descent**

In order to produce the data for this problem, we will apply a probabilistic interpretation of linear regression. We will then use stochastic gradient descent to optimize the cost.

1. Data sampling of 1 million data points is done in code
2. **and c)**

**Original Theta = [3, 2, 1]**

**Batch size = 1:**

The parameters are updated when the batch size is one by analyzing the gradient of the cost function for each data sample. Thus, each iteration of the algorithm involves 1 million steps. Since we only perform update with each batch of one sample without going through the complete dataset, the convergence is incredibly quick.

**Iterations taken to converge: 1**

**Stopping criteria: current cost - previous cost < 4e-4**

**Time Taken: 6.4314796924591064 sec**

**For Learned model, error on new test data: 0.9908188085081633**

**For Original model, error on new test data: 0.9829469215**

**The difference of error in original and learned model for new test data is: 0.007871887008163325**

**Batch size = 100:**

The path of gradient descent of the parameter for a batch size of 100 samples. So the update is based on 100 samples at a time.

**Iterations taken to converge: 2**

**Stopping criteria: current cost - previous cost < 1e-7**

**Time Taken: 9.673362255096436 sec**

**For Learned model, error on new test data: 0.9837823600531629**

**For Original model, error on new test data: 0.9829469215**

**The difference of error in original and learned model for new test data is: 0.000835438553162926**

**Batch size = 10000:**

The path of the theta convergence for a batch size of 10000 out of a 1 million sample space. So the update is based on 10000 samples at a time.

**Iterations taken to converge: 135**

**Stopping criteria: current cost - previous cost < 1e-7**

**Time Taken: 371.48295068740845 sec**

**For Learned model, error on new test data: 0.998534217467866**

**For Original model, error on new test data: 0.9829469215**

**The difference of error in original and learned model for new test data is: 0.01558729596**

**Batch size = 1000000:**

The path of the theta convergence for a batch size of 1 million sample. So the update is based on 1 million samples at a time.

**Iterations taken to converge: 1828**

**Stopping criteria: current cost - previous cost < 1e-8**

**Time Taken: 11912.89756374 sec**

**For Learned model, error on new test data: 0.9805581310521531**

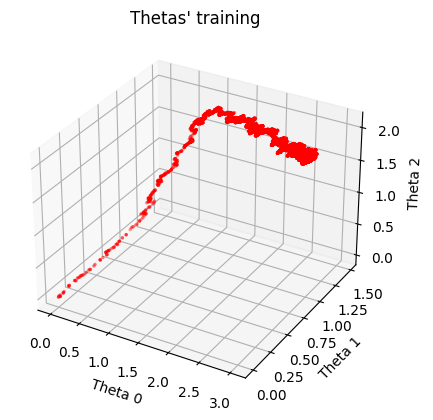
**For Original model, error on new test data: 0.9829469215**

**The difference of error in original and learned model for new test data is: 0.0023887904**

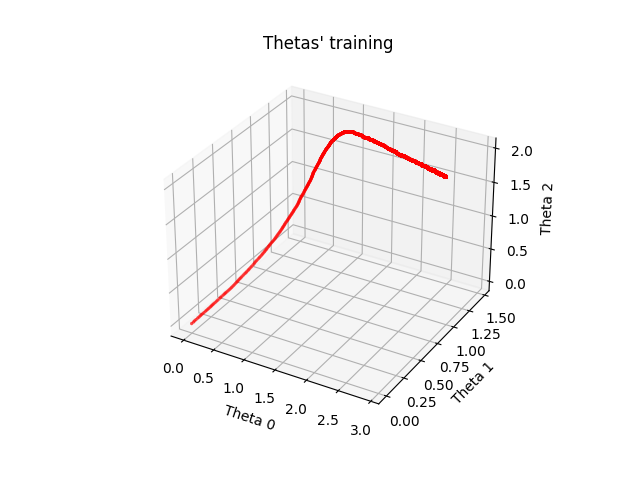
For Learning Rate = 0.001, theta converges in all four cases to values that are quite similar to one another but not precisely the same because we chose different stopping criteria for each of the four situations. And these Theta Parameter values are extremely similar to the original Theta values that we used to sample the data.

Also as the batch size increases convergence time also increases and time complexity approaches O(m\*n) where m is number of examples.

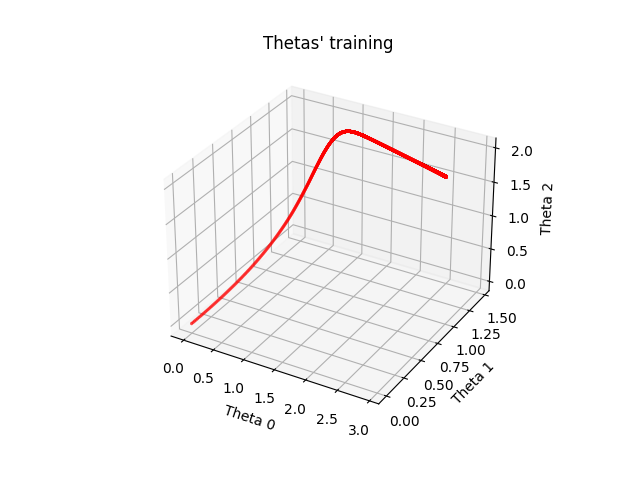
**d) 3D Plots of movement:**



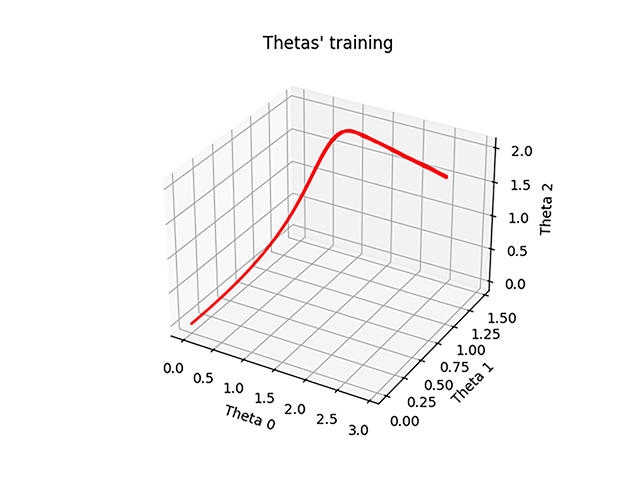
**Batch size = 1**



**Batch size = 100**



**Batch size = 10000**



**Batch size = 1000000**

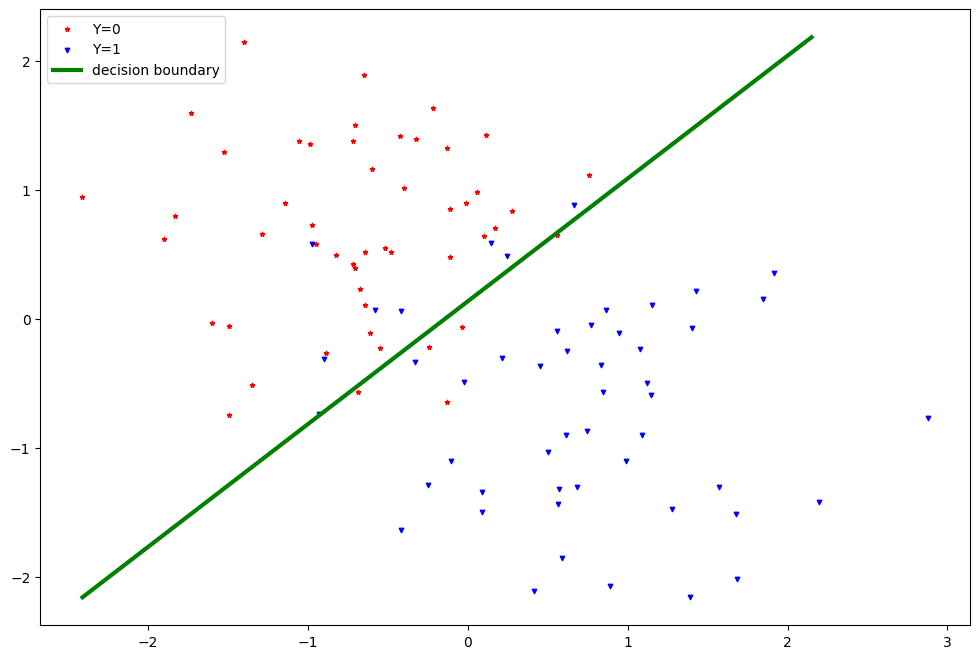
As it is clearly seen in the plots that the path of Theta convergence become less random as we increases the batch size. And it make sense because SGD update theta with randomly chosen data samples and smaller the batch size increases the randomness of Theta convergence path.

**3) Logistic Regression:**

Logistic Regression is a classification model used to classify different classes of data. In this section, we use the Newton’s Method for optimizing the negative log likelihood of the cost function of logistic regression.

1. final parameter obtained from this optimizing step is:

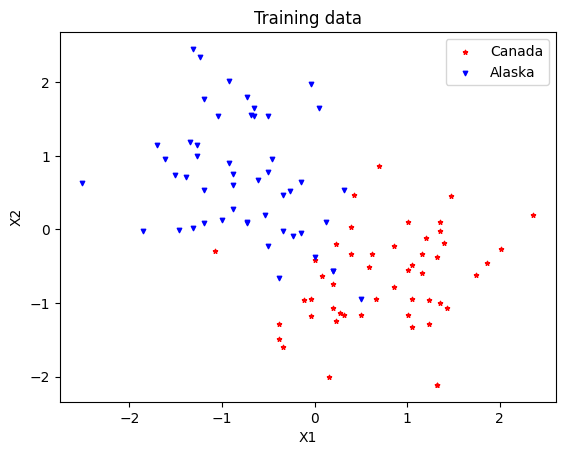
= [0.35211649, 2.44761566, -2.57164248]



**4) Gaussian Discrmimant Analysis:**

1. We assume that both the classes have the same Covariance. So we get a linear separator. The values of the mean and covariance are:

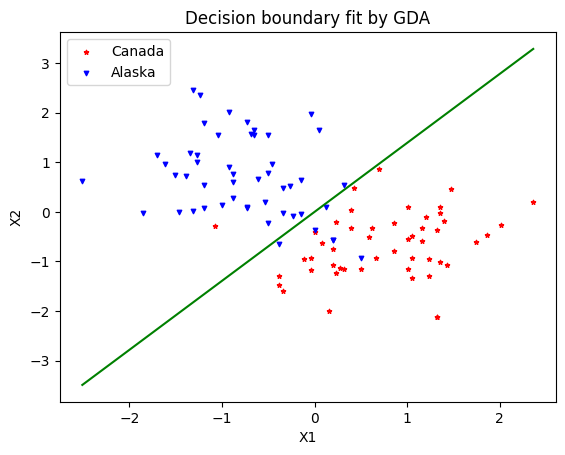
= = ∑ =



**Training Data**

1. Equation of Decision Boundary for = = ∑

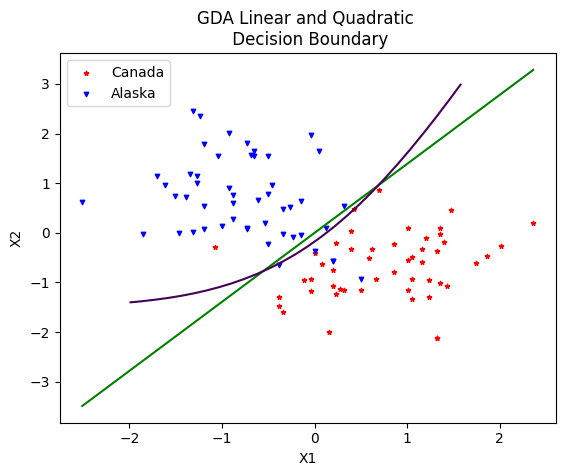
= 0



1. **, , ,**

1. Equation for the quadratic boundary separating the two regions:

= 0



1. Observations:

The GDA's learned linear border is not the best way to divide the two groups. It can classify the data with high accuracy but the data is not well separated by this linear boundary. So it unlikely to predict the class of new data point. It would prevent us from classifying points with a high degree of accuracy. This occurs as a result of the two classes' absence of covariance information. As a result, it is comparable to logistic regression. However, the quadratic decision boundary effectively divides the data because it is a good data separator since it understands the correlation between the two classes.